# 0SASS用法指南

作者： [阮一峰](http://www.ruanyifeng.com/)

日期： [2012年6月19日](http://www.ruanyifeng.com/blog/2012/06/)

学过[CSS](http://zh.wikipedia.org/wiki/%E5%B1%82%E5%8F%A0%E6%A0%B7%E5%BC%8F%E8%A1%A8" \t "/home/jason/文档\\x/_blank)的人都知道，它不是一种编程语言。

你可以用它开发网页样式，但是没法用它编程。也就是说，CSS基本上是设计师的工具，不是程序员的工具。在程序员眼里，CSS是一件很麻烦的东西。它没有变量，也没有条件语句，只是一行行单纯的描述，写起来相当费事。



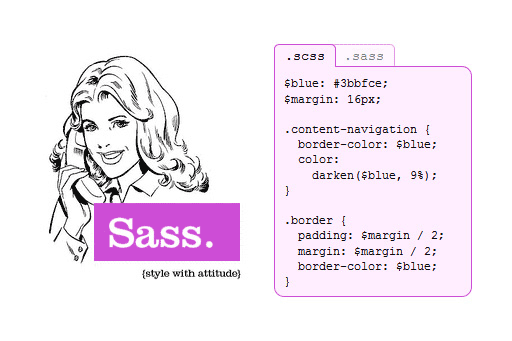
很自然地，有人就开始为CSS加入编程元素，这被叫做["CSS预处理器"](http://www.catswhocode.com/blog/8-css-preprocessors-to-speed-up-development-time" \t "/home/jason/文档\\x/_blank)（css preprocessor）。它的基本思想是，用一种专门的编程语言，进行网页样式设计，然后再编译成正常的CSS文件。

各种"CSS预处理器"之中，我自己最喜欢[SASS](http://sass-lang.com/" \t "/home/jason/文档\\x/_blank)，觉得它有很多优点，打算以后都用它来写CSS。下面是我整理的用法总结，供自己开发时参考，相信对其他人也有用。

============================================

****SASS用法指南****

作者：阮一峰



****一、什么是SASS****

[SASS](http://sass-lang.com/" \t "/home/jason/文档\\x/_blank)是一种CSS的开发工具，提供了许多便利的写法，大大节省了设计者的时间，使得CSS的开发，变得简单和可维护。

本文总结了SASS的主要用法。我的目标是，有了这篇文章，日常的一般使用就不需要去看[官方文档](http://sass-lang.com/docs/yardoc/file.SASS_REFERENCE.html" \t "/home/jason/文档\\x/_blank)了。

****二、安装和使用****

****2.1 安装****

SASS是Ruby语言写的，但是两者的语法没有关系。不懂Ruby，照样使用。只是必须先[安装Ruby](http://www.ruby-lang.org/zh_cn/downloads/" \t "/home/jason/文档\\x/_blank)，然后再安装SASS。

假定你已经安装好了Ruby，接着在命令行输入下面的命令：

　　gem install sass

然后，就可以使用了。

****2.2 使用****

SASS文件就是普通的文本文件，里面可以直接使用CSS语法。文件后缀名是.scss，意思为Sassy CSS。

下面的命令，可以在屏幕上显示.scss文件转化的css代码。（假设文件名为test。）

　　sass test.scss

如果要将显示结果保存成文件，后面再跟一个.css文件名。

　　sass test.scss test.css

SASS提供四个[编译风格](http://sass-lang.com/docs/yardoc/file.SASS_REFERENCE.html" \l "output_style" \t "/home/jason/文档\\x/_blank)的选项：

　　\* nested：嵌套缩进的css代码，它是默认值。

　　\* expanded：没有缩进的、扩展的css代码。

　　\* compact：简洁格式的css代码。

　　\* compressed：压缩后的css代码。

生产环境当中，一般使用最后一个选项。

　　sass --style compressed test.sass test.css

你也可以让SASS监听某个文件或目录，一旦源文件有变动，就自动生成编译后的版本。

　　// watch a file

　　sass --watch input.scss:output.css

　　// watch a directory

　　sass --watch app/sass:public/stylesheets

SASS的官方网站，提供了一个[在线转换器](http://sass-lang.com/try.html" \t "/home/jason/文档\\x/_blank)。你可以在那里，试运行下面的各种例子。

****三、基本用法****

****3.1 变量****

SASS允许使用变量，所有变量以$开头。

　　$blue : #1875e7;

　　div {  
　　　color : $blue;  
　　}

如果变量需要镶嵌在字符串之中，就必须需要写在#{}之中。

　　$side : left;

　　.rounded {  
　　　　border-#{$side}-radius: 5px;  
　　}

****3.2 计算功能****

SASS允许在代码中使用算式：

　　body {  
　　　　margin: (14px/2);  
　　　　top: 50px + 100px;  
　　　　right: $var \* 10%;  
　　}

****3.3 嵌套****

SASS允许选择器嵌套。比如，下面的CSS代码：

　　div h1 {  
　　　　color : red;  
　　}

可以写成：

　　div {  
　　　　hi {  
　　　　　　color:red;  
　　　　}  
　　}

属性也可以嵌套，比如border-color属性，可以写成：

　　p {  
　　　　border: {  
　　　　　　color: red;  
　　　　}  
　　}

注意，border后面必须加上冒号。

在嵌套的代码块内，可以使用&引用父元素。比如a:hover伪类，可以写成：

　　a {  
　　　　&:hover { color: #ffb3ff; }  
　　}

****3.4 注释****

SASS共有两种注释风格。

标准的CSS注释 /\* comment \*/ ，会保留到编译后的文件。

单行注释 // comment，只保留在SASS源文件中，编译后被省略。

在/\*后面加一个感叹号，表示这是"重要注释"。即使是压缩模式编译，也会保留这行注释，通常可以用于声明版权信息。

　　/\*!   
　　　　重要注释！  
　　\*/

****四、代码的重用****

****4.1 继承****

SASS允许一个选择器，继承另一个选择器。比如，现有class1：

　　.class1 {  
　　　　border: 1px solid #ddd;  
　　}

class2要继承class1，就要使用@extend命令：

　　.class2 {  
　　　　@extend .class1;  
　　　　font-size:120%;  
　　}

****4.2 Mixin****

Mixin有点像C语言的宏（macro），是可以重用的代码块。

使用@mixin命令，定义一个代码块。

　　@mixin left {  
　　　　float: left;  
　　　　margin-left: 10px;  
　　}

使用@include命令，调用这个mixin。

　　div {  
　　　　@include left;  
　　}

mixin的强大之处，在于可以指定参数和缺省值。

　　@mixin left($value: 10px) {  
　　　　float: left;  
　　　　margin-right: $value;  
　　}

使用的时候，根据需要加入参数：

　　div {  
　　　　@include left(20px);  
　　}

下面是一个mixin的实例，用来生成浏览器前缀。

　　@mixin rounded($vert, $horz, $radius: 10px) {  
　　　　border-#{$vert}-#{$horz}-radius: $radius;  
　　　　-moz-border-radius-#{$vert}#{$horz}: $radius;  
　　　　-webkit-border-#{$vert}-#{$horz}-radius: $radius;  
　　}

使用的时候，可以像下面这样调用：

　　#navbar li { @include rounded(top, left); }

　　#footer { @include rounded(top, left, 5px); }

****4.3 颜色函数****

SASS提供了一些内置的颜色函数，以便生成系列颜色。

　　lighten(#cc3, 10%) // #d6d65c  
　　darken(#cc3, 10%) // #a3a329  
　　grayscale(#cc3) // #808080  
　　complement(#cc3) // #33c

****4.4 插入文件****

@import命令，用来插入外部文件。

　　@import "path/filename.scss";

如果插入的是.css文件，则等同于css的import命令。

　　@import "foo.css";

****五、高级用法****

****5.1 条件语句****

@if可以用来判断：

　　p {  
　　　　@if 1 + 1 == 2 { border: 1px solid; }  
　　　　@if 5 < 3 { border: 2px dotted; }  
　　}

配套的还有@else命令：

　　@if lightness($color) > 30% {  
　　　　background-color: #000;  
　　} @else {  
　　　　background-color: #fff;  
　　}

****5.2 循环语句****

SASS支持for循环：

　　@for $i from 1 to 10 {  
　　　　.border-#{$i} {  
　　　　　　border: #{$i}px solid blue;  
　　　　}  
　　}

也支持while循环：

　　$i: 6;

　　@while $i > 0 {  
　　　　.item-#{$i} { width: 2em \* $i; }  
　　　　$i: $i - 2;  
　　}

each命令，作用与for类似：

　　@each $member in a, b, c, d {  
　　　　.#{$member} {  
　　　　　　background-image: url("/image/#{$member}.jpg");  
　　　　}  
　　}

****5.3 自定义函数****

SASS允许用户编写自己的函数。

　　@function double($n) {  
　　　　@return $n \* 2;  
　　}

　　#sidebar {  
　　　　width: double(5px);  
　　}

（完）